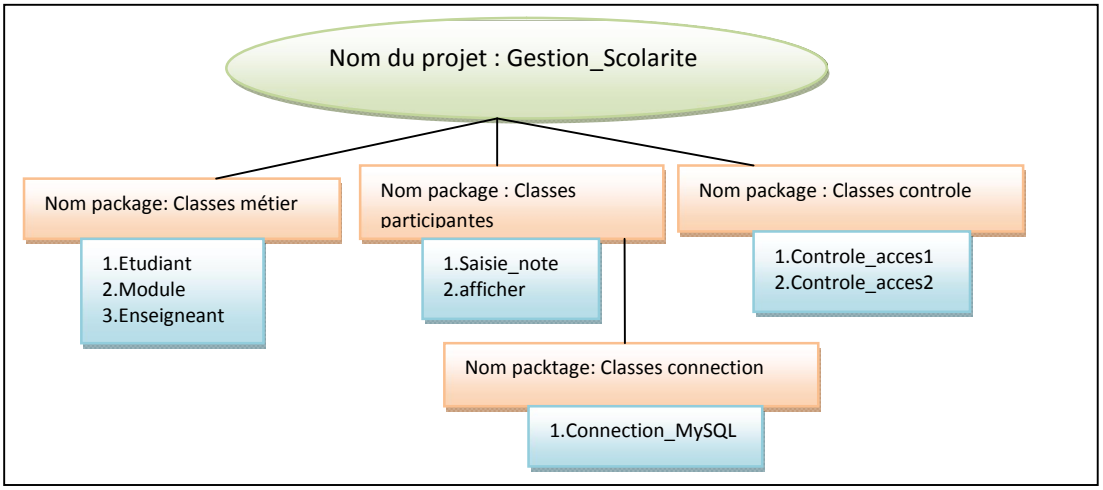
TP N01
(Création des package)

Enoncé : une application de gestion de scolarité intitulé gestion_scolarite comporte 4 packages et 8 classes selon l'arborescence suivante :



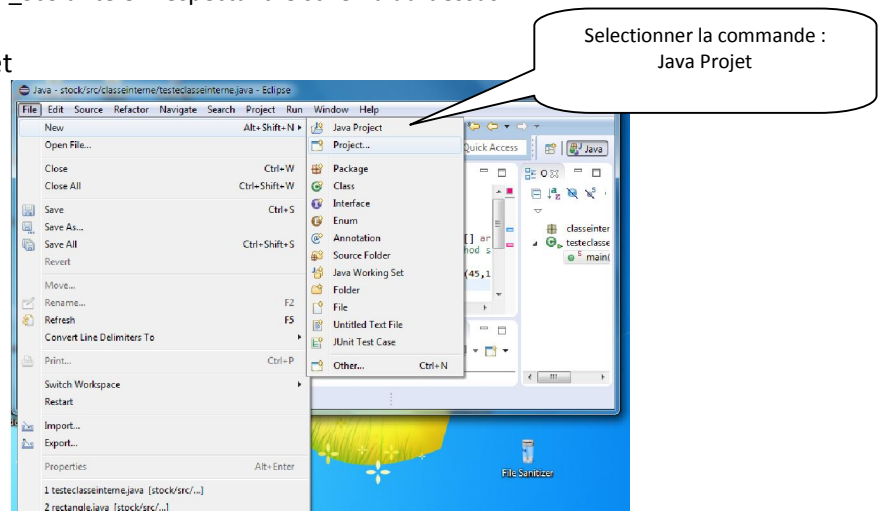
Travail à faire :

I. lancer IDE Eclipse

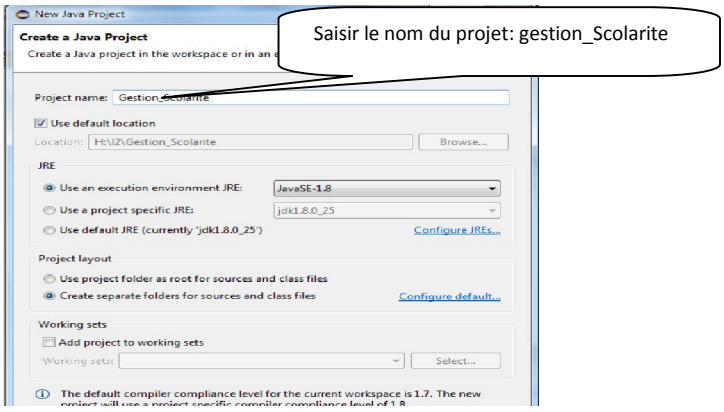
II. créer le projet Gestion_Scolarite en respectant le schéma au-dessus

<u>Réponse</u> :

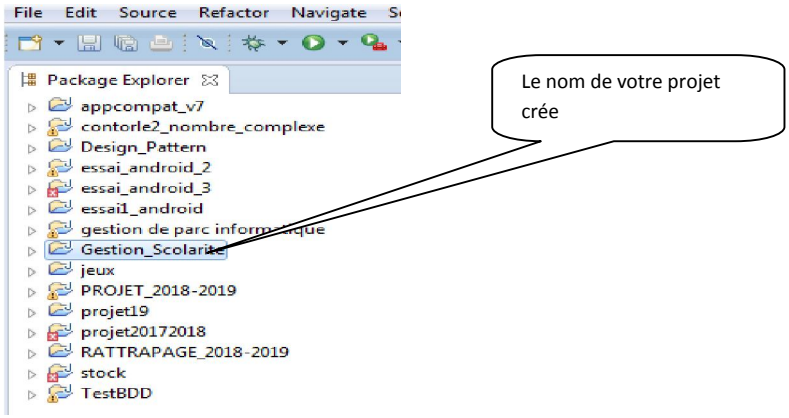
1-pour créer votre projet



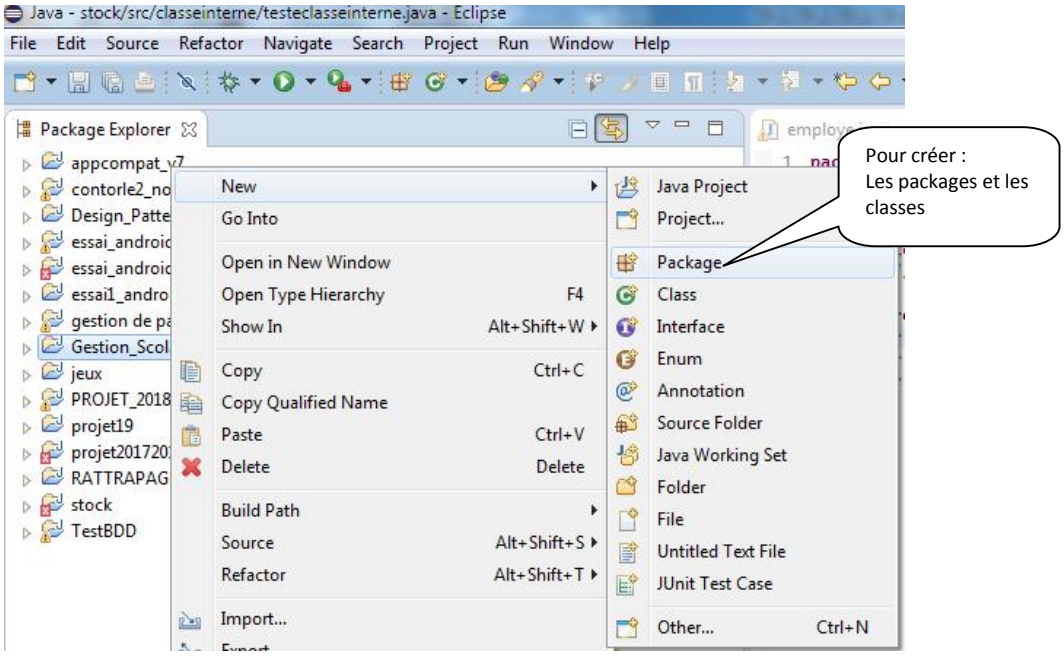
2-saisir le nom du projet



3.votre projet est affiché sur le volet package Explorer



4-en cliquant avec le bouton droit de la souris sur votre projet, vous allez avoir les commandes permettant de créer les packages et les classes associées :



Vous pouvez continuer le travail …….

TP N02
(les modificateurs de visibilité)

Enoncé :

1-créer un nouveau projet sous le nom TP2_encapsulation

2-créer la classe suivante :

```
public class Rectangle {
        private    int largeur;
        protected  int longueur;
        public     int lon;
                   int lar;
        public     Rectangle(int  x,int y){this.largeur=x;this.longueur=y;}
        public     void affiche(){System.out.println("largeur "+largeur+" "+"longueur"+longueur+"
lon====="+lon+"   lar=====  "+lar );}}
```

3- créer la classe de test suivante :

```
public class Test_Rect{
        public static void main(String[] args) {
                // TODO Auto-generated method stub
                                Rectangle r=new Rectangle (45,15);
                r.afficher();}}
```

4-exécuter la classe de test et expliquer le résultat affichés
5-ajouter le code suivant dans la classe de test

```
r.lon=50;
r.lar=60;
r.affiche();
r.longueur=12;
r.largeur=0;
```

6-exécuter la classe de test et expliquer le résultat affichés
7-corriger les erreurs
8-exécuter la classe de test
9-écrire un paragraphe traduisant le concept de l'encapsulation sur votre projet

TP N03
(Les niveaux de visibilité)

Indication :

- des erreurs seront signalées par le **compilateur (une ligne qui contient une erreur le compilateur l'indique à gauche par un petit rectangle en couleur rouge avec une croix à l'intérieur)**
- nous allons introduire la notion de l'héritage par le mot réservé **extends** (voir chapitre suivant) dans ce TP pour monter le niveau de visibilité.
- Au cours de la réalisation du présent TP. On vous demande à chaque étape de dégager et d'expliquer les erreurs.

Enoncé :
1-créer un projet sous le nom TP03_encapsulation
2-créer deux packages P1 et P2 avec les programmes associés selon le schéma suivant :

```
package p1 ;
class C1 {
public  int a=10;
protected int b=20;
int c=30;
private int d=50;
static int m=5;}
class C2 extends C1 {}
class C3 {}
```

```
package p2 ;
class C4 extends C1 {}
class C5 {}
```

3-ajouter le code suivant dans la classe C2

```
int e=a+15;
int f=b+12;
int j=c+47;
int u=d+2;
```

4- dégager et expliquer les erreurs ?
3-ajouter le code suivant dans la classe C3

```
C1 t =new C1();
int o=t.a;
int p=t.b;
```

```
int l=t.c;
int i=t.m;
int j=C1.m;
```

5- est ce qu'il y a des erreurs, expliquer la situation ?

6- ajouter le code suivant dans la classe C4

```
C1 c11=new C1();
int e=c11.a+1;
int f=b+1;
int h=c+12;
```

7-expliquer les erreurs ?

8-ajouter le code suivant dans la classe C5

```
C1 t =new C1();
 int o=t.a+45;
int k=t.b+5;
int p=t.c+1;
```

9- dégager et expliquer les erreurs ?

10-remplir le tableau suivant par **Y** lorsque la variable est accessible et par **N** dans le cas contraire :

	Variable (a)	Variable (b)	Variable (c)	Variable (d)	Variable (m)
l'accessibilité de C2					
l'accessibilité de C3					
l'accessibilité de C4					
l'accessibilité de C5					

11-écrivez une classe de test qui contient la méthode main() pour afficher le contenu de toutes les variables d'instance (a,b,c,d…..) et la variable de classe (`static int m=5`)

Réponse : suivre les étapes et vous allez trouvez les mêmes résultats expliquées dans le chapitre 2 : Encapsulation (page 6)

TP N04

(Les Accesseurs setters and getters)

Indication :

- Le développement des squelettes des constructeurs, setters et getters peut être génères automatiquement par l'Eclipse, comme on peut les développer nous même.

Enoncé :

1-Créer un projet sous le nom TP04_encapsulation avec un package sous le nom Set_Get

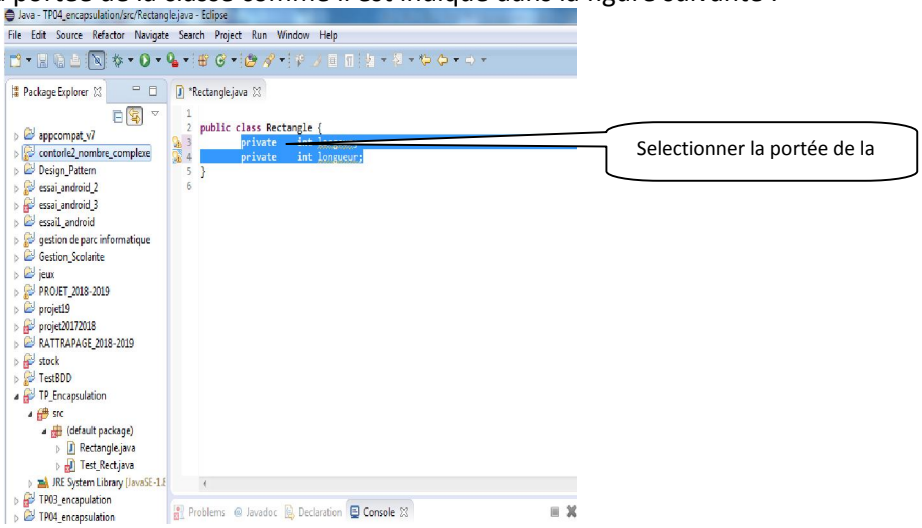
2-écrivez la classe Rectangle suivante :

```
public class Rectangle {
      private    int largeur;
      private    int longueur; }
```

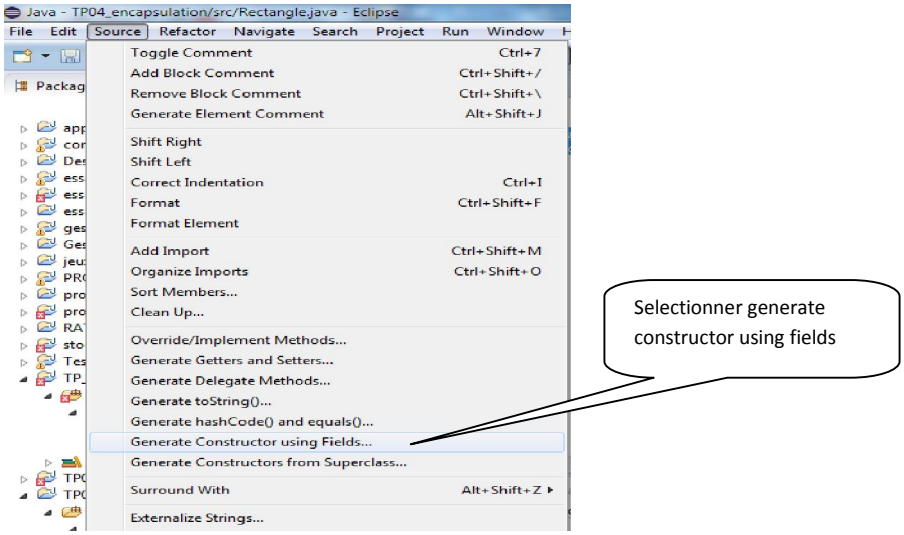
a-génération automatique de constructeur,setters et getters

génération du constructeur

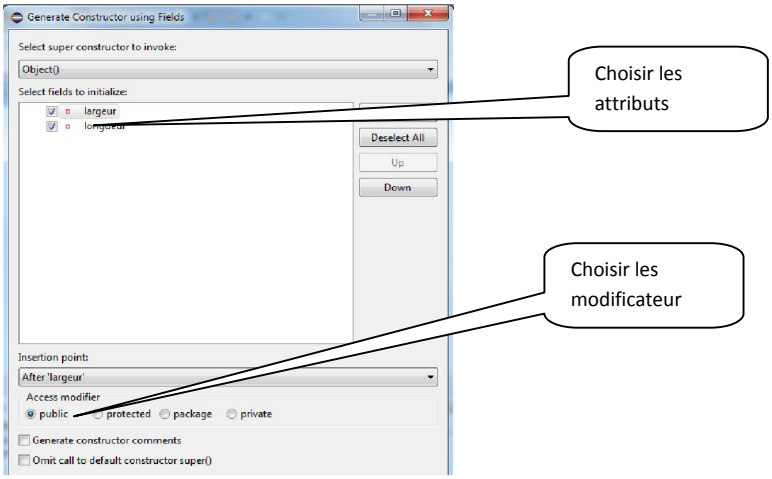
a-1- selectionner la portée de la classe comme il est indiqué dans la figure suivante :



a-2- aller au menu Source en validant la commande Generate Constructor using Fields ,comme il est indiqué dans la figure suivante :



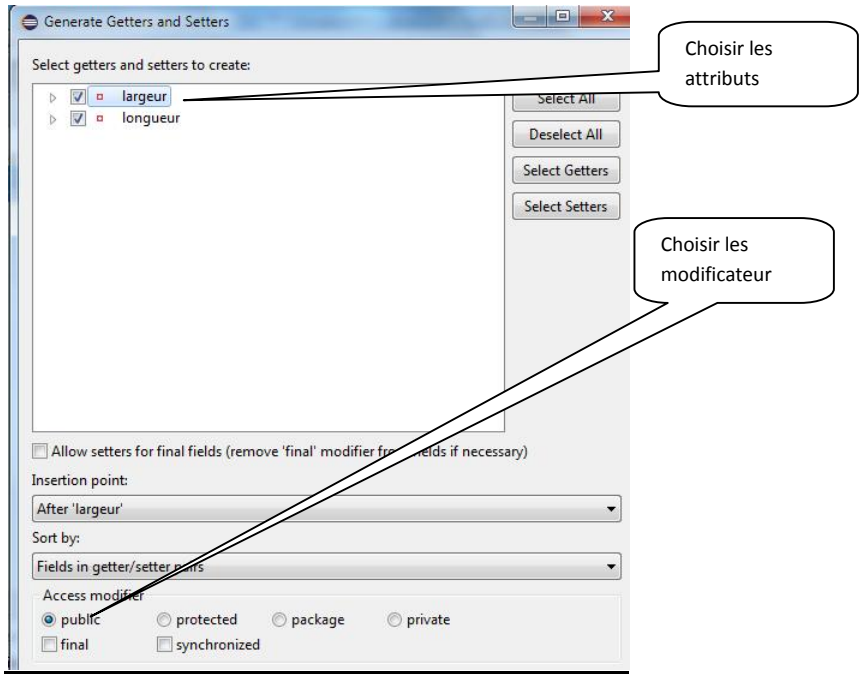
a-3- choisir les attributs et leur modificateur d'accès pour votre constructeur, comme il est indiqué dans la figure suivante :



a-4- valider le bouton OK (un constructeur paramétré avec les deux attributs sera généré automatiquement)

génération du setters et getters

a-1- A partir de l'étape a-2- de génération du constructeur en validant la commande Generate Getters and Setters ,vous allez avoir la situation suivante :



a-4- valider le bouton OK : quatre méthodes seront générées deux : set et get pour longueur set et get pour largeur)

Remarque : la méthode automatique reste pauvre le fait qu'elle génère que le squelette des méthodes. Donc reste au programmeur de compléter le code s'il y a lieu.

la méthode manuelle :

Elle est réalisée par le programmeur qui reste toujours la plus adéquate, le fait que le programmeur a le droit complet pour gérer son code.

Travail à faire : créer une classe de test pour exécuter la classe Rectangle

TP N05
(l'objet This)

Enoncé : un Cercle est définit par un rayon , on peut calculer sa surface et son périmètre.
En utilisant l'objet **This**,créer un projet sous le nom TP05_encapsulation qui va définir la classe Cercle avec:
a-Deux constructeurs, l'un est paramétré par le paramètre rayon et l'autre sans paramètre mais fait appel au premier constructeur pour créer un objet de rayon égale à 0 .
b-les setters et getters
c-les deux méthode permettant de calculer la surface et le périmètre d'un cercle
d-une méthode affiche qui permet d'afficher les caractéristiques d'un objet de type cercle
II.2 créer une classe de test sous le nom Test_TP05 qui contient la méthode main

II.3 instancier la classe Cercle en créant plusieurs objets Cercle
II.4 exécuter le projet pour plusieurs cas de rayon

Réponse

Constructeurs

```
public class Cercle {
	int rayon;
	Cercle(int rayon){	this.rayon=rayon;}// this est utilisé comme un objet}
	Cercle() { this(0);  }// this est utilisé comme une méthode

}
```

Les setters et getters

```
public int getRayon() {
		return rayon;
	}
	public void setRayon(int rayon) {
		this.rayon = rayon;
	}
```

Les deux méthode surface et périmètre

```
public int surface () { compléter le code….  }
	public void peremetre() { compléter le code….}
```

La méthode affiche

```
public	void affiche(){System.out.println("……compléter le code….)}
```

……. A compléter et afficher les résultats sur la console

TP N06
(Variables et méthodes de classe (static))

Enoncé :

Dans un magasin, un article est caractérisé par le code, le prix et le nom. On voudra compter le nombre des articles dans le magasin (prévoir une variable statique pour l'incrémenter lors de chaque instanciation de la classe Article) et une méthode statique pour afficher le nombre des article stockés dans le magasin.

Travail à faire :

1-créer la classe Article en répondant aux besoins de l'énoncé du TP
2-tester la classe Article

Réponse

```
 public class Article {

private int code;
private float prix;
private String nom;
private static int  cpt;
public Article(int code,String nom, float prix) {
      this.code = code;this.nom = nom;this.prix = prix;
      cpt++;
}
public int getCode() {    return code;}
public void setCode(int code) { this.code = code;}
public float getPrix() {  return prix;}
public void setPrix(float prix) {      this.prix = prix;}
public String getNom() {  return nom;}
public void setNom(String nom) {this.nom = nom;}
public     void affiche(){System.out.println("code "+code+" "+"nom  "+nom+"
prix"+prix);}

public static void nombre_article(){System.out.println(cpt);}


}
```

Tester la classe :

```
public class Test_TP06_encapsulation {
      public static void main(String[] args) {
            // TODO Auto-generated method stub
                        Article r=new Article (45,"computer",15);
            r.affiche();
            Article.nombre_article();// appel a la methode statique
            r.nombre_article();// appel a la methode statique
            }     }
```